논문 2005-42SD-2-9

효율적인 다중 채널 On-Chip-Bus를 위한 SoC Network Architecture

(SoC Network Architecture for Efficient Multi-Channel On-Chip-Bus)

이 상 헌*, 이 찬 호**, 이 혁 재***

(Sanghun Lee, Chanho Lee, and Hyuk-Jae Lee)

요 약

공정기술과 EDA 툴의 발전에 따라서 하나의 실리콘 다이(Die)에 많은 IP가 집적되고 멀티프로세서가 포함되는 SoC 구조가 가능해지고 있다. 그러나 대부분의 기존 SoC 버스는 공유버스 구조라는 문제점으로 인해 통신의 병목현상이 발생하고 이는 전체 시스템 성능을 저하시키는 요인이 된다. 많은 경우에 멀티프로세서 시스템의 성능은 CPU 자체의 속도보다는 효율적인 통신과 균형있는 연산의 분배가 좌우하게 된다. 따라서 충분한 SoC 버스 대역폭(Bandwidth)을 확보하기 위한 하나의 해결책으로 크로스바 라우터(Crossbar Router)를 이용하여 효율적인 온 칩 버스구조인 SoC Network Architecture(SNA)를 제안한다. 제안된 SNA구조는 다중 마스터(multi-master)에 대해 다중 채널(multi-channel)을 제공하여 통신의 병목현상을 크게 줄일 수 있으며 뛰어난 확장성을 지원한다. 제안된 구조에 따라 모델 시스템을 설계하고 시뮬레이션을 진행한 결과 AMBA AHB 버스에 비해 평균 40% 이상 효율이 증가했다.

Abstract

We can integrate more IP blocks on a silicon die as the development of fabrication technologies and EDA tools. Consequently, we can design complicated SoC architecture including multi-processors. However, most of existing SoC buses have bottleneck in on-chip communication because of shared bus architectures, which result in the performance degradation of systems. In most cases, the performance of a multi-processor system is determined by efficient on-chip communication and the well-balanced distribution of computation rather than the performance of the processors. We propose an efficient SoC Network Architecture(SNA) using crossbar routers which provide a solution to ensure enough communication bandwidth. The SNA can significantly reduce the bottleneck of on-chip communication by providing multi-channels for multi-masters. According to the proposed architecture, we design a model system for the SNA. The proposed architecture has a better efficiency by 40% than the AMBA AHB according to a simulation result.

Keywords : on-chip-bus, SoC bus, on-chip-network, crossbar router, switch matrix

I. 서 론

공정기술과 EDA 툴의 발전에 따라서 하나의 실리콘 다이(die) 에서 보다 많은 IP 블록의 통합이 기술적으로 가능하게 되었다. 이런 기술의 발달로 멀티미디어와 통신 등 연산 작업량이 많은 곳에서 필요로 하는 병렬 처리 연산이 요구되고 있다.

멀티프로세서 SoC (System-on-Chip)에서는 상호 통신구조(communication architecture)에서 병목현상이 발생하며, 공유 통신 자원에 대한 접근을 동시에 요청할 때 시스템의 성능 저하를 막을 수 있는 효율적인 방법이 필요하게 된다. 이러한 문제는 기존의 SoC 버스가 공유버스(shared bus)구조를 가짐으로써 병목현상이 발생하기 때문이다.

* 학생회원, 숭실대학교 전자공학과
(Dept. of Electronic Engr., Soongsil University)
** 정회원, 숭실대학교 정보통신전자공학부
(School of Electronic Engr., Soongsil University)
*** 정회원, 서울대학교 전기공학부
(School of Electrical Engr., Seoul National University)
※ 본 연구는 한국과학재단 목적기초연구(R01-2003-000-10582-0 (2003))지원으로 수행되었음.
접수일자: 2004년7월2일, 수정완료일: 2005년1월26일

현재 세계 시장의 70% 이상을 차지하고 있는 ARM 사의 AMBA는[1] 다른 버스에 비해서 훨씬 간단한 프로토콜을 가지기 때문에 소규모 SoC에서는 비교적 쉽게 사용할 수가 있었다. 이에 따라 많은 IP 벤더들이 AMBA 호환 IP를 개발하여 SoC 시장에서 AMBA는 온 칩 버스의 대표자리를 굳혀 왔으나, 이처럼 간단한 버스 프로토콜은 SoC의 규모가 방대해지면서 그 효율성의 한계를 드러내고 있다. 즉, AMBA는 단순한 SoC 설계에는 적합하지만 복잡한 SoC용으로는 그 기능이 부족한 것이다. 따라서 AMBA의 문제점을 극복하는 새로운 SoC 버스 표준의 필요성이 절실하다. 그러나 기존의 무수한 IP들이 AMBA를 지원하기 때문에 이를 버리고 새로운 온 칩 버스를 채택하기 위해서는 기존의 IP를 다시 설계해야 하는 어려운 점도 많이 있다. 그 밖의 AMBA와 유사한 목적으로 개발된 기존의 온 칩 버스들은 대부분 버스 대역폭이 고정되어 있기 때문에 새로운 IP가 추가되면 하나의 IP가 사용가능한 버스 대역폭은 줄어들기 쉽다.

또한 기존 SoC 버스에서는 대역폭의 병목 등의 이유로 하나의 버스에서 사용될 수 있는 마스터의 수를 제한하였다. 더 많은 마스터를 사용하기 위해서는 복잡한 계층구조(hierarchical topology)를 갖게 된다. 개발자는 이러한 부분에 익숙하지 않기 때문에 프로토콜에 익숙해지기 위해서는 많은 시간이 필요하게 된다. 또한 계층구조를 사용했을 때 브리지를 거쳐서 통신을 해야 하기 때문에 데이터의 전달지연(propagation latency)이 증가하게 되며 하나의 집중화된 브리지(centralized bridge)는 공유버스 문제 같은 병목을 발생시키는 요인이 될 수 있다.

한편, 집적도 증가에 따라 온 칩 버스는 실제 배치(Layout)상에서 아주 긴 와이어가 되는데 이 경우 와이어간 간섭 등의 문제로 지연, 오류의 발생 및 전력 소모 증가가 우려된다. 또한 기존 SoC 버스는 tri-state 버퍼를 사용하지 않고 공유버스를 구현하기 위해 일반적으로 multiplexer 기반의 구조를 갖고 있다. 이 구조의 특징은 여러 슬레이브에 신호를 전달하기 위해 broadcasting 방법을 사용하는 것이다. 필요 이상의 슬레이브에 broadcasting 하기 때문에 이때 구동해야하는 capacitance의 증가로 추가적인 에너지소모가 발생하게 된다.

현실적인 대안으로써 연구되고 있는 NoC (Network on Chip)의 경우도 아직까지 구현 형태가 지나치게 복잡하고 중재 알고리즘 처리를 위한 계산량이 많아 오히려 전력 소모 증가와 성능 저하를 가져올 수 있다. 이는 기존의 슈퍼컴퓨터나 서버 제작을 위한 네트워크 구현법을 그대로 모방하기 때문에 생기는 문제이다.[2] Gurrier 등은 버스와 네트워크 구조를 사용할 때 장/단점에 대해서 언급했는데 네트워크 구조를 사용할 때 내부구조의 특성상 추가적인 잠복기(latency)가 발생하며, 상당한 실리콘 면적을 차지하게 된다는 것을 주요 단점으로 지적했다.[3]

대부분의 기존 SoC 버스구조는 중재 토폴로지(topology)를 갖는 다중 버스 분할 연결을 지원하고 동시에 적절한 확장성을 지원하도록 설계되었다. AMBA [4], Wishbone [5], CoreConnect [6] 등이 적절한 예로 볼 수 있다. Wishbone, CoreConnect, AMBA는 중재정책 및 추가적인 몇몇 특징 이외에는 모두 공유버스 구조를 가지고 있기 때문에 AMBA 버스와 함께 다른 버스들도 동일한 문제점을 갖고 있다.

따라서 이러한 문제점들을 해결하기 위해서 본 논문에서 단순한 구조를 가지면서 동시에 다중 마스터에 대해서 다중 채널을 제공함으로써 대역폭을 확보하며, 버스에 추가되는 IP의 증가를 고려하여 확장이 용이한 SNA(SoC Network Architecture)를 제안한다.

SNA는 주소와 대부분의 제어 신호를 데이터 선에 포함시키고 최소한의 추가 제어 신호선을 갖도록 하여 연결선의 수를 대폭 줄인 SNP(SoC Network Protocol)를[7] 인터페이스 프로토콜로 사용한다.

SNA 내부의 물리적 구성요소는 4N개의 채널을 갖는 크로스바 라우터(Crossbar Router)와 이들의 상호연결을 중재하는 광역중재기(Global Arbiter), 그리고 IP 또는 sub-system을 크로스바 라우터와 연결시키는 스위치 래퍼/브리지(Switch Wrapper/Bridge)로 구성된다. 이들 구성요소가 필요에 따라 적절한 토폴로지로 구성되어 다중 마스터에 대해서 다중 채널을 제공하는 SoC용 버스를 형성한다. SNA는 스위치 래퍼/브리지를 통해 기존 AMBA 버스와의 호환성을 유지하고 크로스바 라우터의 적절한 배치를 이용한 유연한 구조를 가지고 있으며, 세 가지의 동작모드를 통해 잠복기를 최소화하는 특징을 가지고 있다. 본 논문에서 SNA 구조를 제안하고 하나의 모델 시스템을 설계하여 모의실험을 통해 동작을 검증하였다.

Ⅱ. 제안된 버스구조

SNA는 SoC내에서 효율적인 버스사용을 지원하기

위한 구조이다. 기존의 버스는 공유버스구조를 바탕으로하여 다중 프로세서(다중 마스터)구조에서 버스의 통신 병목현상으로 효율이 저하되는 문제점이 있다. SNA는 이를 극복하기 위하여 다중 채널을 제공하고 다중 마스터가 채널에 접근하여 동시에 통신을 할 수 있는 구조를 가지고 있다. SNA는 다중 통신채널을 제공하는 크로스바 라우터와 이들의 상호연결을 중재하는 광역중재기(Global Arbiter), 그리고 IP 또는 지역버스를 갖는 sub-system과 크로스바 라우터를 연결하는 스위치 래퍼/브리지로 구성된다.

그림 1은 제안된 SNA의 각 구성요소와 2-라우터 시스템의 전체 구조를 보여주고 있다. 크로스바 라우터(XR)는 크로스바 스위치 매트릭스(XSM)와 크로스바 스위치 콘트롤러(XSC)로 구성되며 다양한 토폴로지를 지원하는 유연성을 가진다. 먼저 XSM은 실제 데이터/주소를 전달하는 채널의 집합체로서 데이터의 흐름은 단방향성을 가지고 XSC의 제어를 받게 된다. XSM는 XR의 4면에 대해 각각 N개의 링크(link)를 갖는다. 하나의 링크는 하나의 IP 또는 sub-system과 연결될 수 있고 단방향이므로 들어오고 나가는 선들이 각각 필요하다. 현재는 SNA에서 SNP를 채택하고 있으므로 입력과 출력에 대해 각각 기본적인 구조의 경우 32 bit 주소/데이터 선과 5 bit 제어신호로 구성되어 하나의 링크는 74 bit로 구성된다. 주소/데이터 선은 구성조건에 따라 8-64 bit로 변경 가능하다. 그리고 한 면에 몇 개의 링크가 있느냐에 따라 전체 선의 수가 결정된다. 현재는 N=2이면 충분하고 SNA의 이해도 쉬우므로 앞으로는 N=2인 경우에 대해 설명하기로 한다. 이때 XSM은 2-4개의 채널을 제공할 수 있다. 필요한 경우 N의 값은 동일한 개념을 갖고 더 증가시킬 수 있다. XSC는 3가지 중요한 기능을 포함하고 있다. 첫째는, 크로스바 라우터 내의 지역통신을 책임지는 지역중재기(local arbitration)의 역할이다. 지역통신은 뒤에서 설명할 지역라우팅(local routing)에 의해 이루어지며 통신을 원하는 마스터와 슬레이브가 동일한 XR에 연결되어 있을 때 XR 자체적으로 채널을 형성하여 통신이 가능하도록 한다. 둘째는, XSM에 라우팅 명령을 내리는 제어기(Controller) 역할이다. 물리적인 신호의 전송은 XSM을 통해 일어나므로 XSM의 어느 링크를 내부적으로 연결시키는가에 대한 지시를 내려 신호의 흐름을 제어한다. 끝으로 지역라우팅으로 해결할 수 없고 광역라우팅(global routing)이 필요한 통신일 경우 광역중재기에 광역라우팅 요청을 하는 역할을 한다. 통신을 원하는

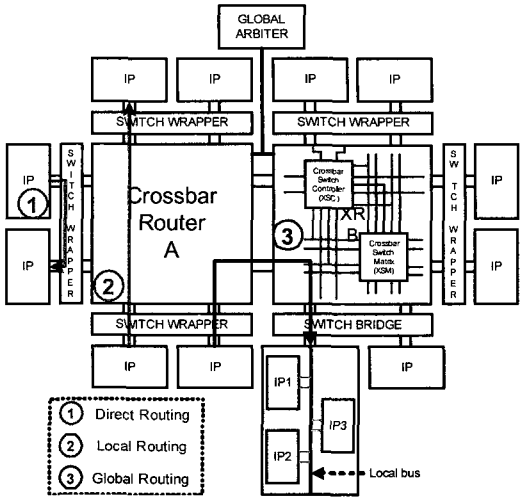

그림 1. 제안된 SNA의 구성요소와 2-라우터 시스템의 구성도

Fig. 1. SNA system with 2 crossbar routers.

마스터와 슬레이브가 서로 다른 XR에 연결되어 있는 경우에는 하나의 XR내의 XSC는 다른 XR의 상태를 알지 못하고 제어할 수도 없으므로 광역중재기를 통해 라우팅 정보를 얻어서 XSM에 대해 제어신호를 발생시키게 된다.

광역중재기(GA)는 두 개 이상의 XR가 사용될 때 필요한 구성요소로서 마스터와 슬레이브가 서로 다른 XR에 연결되어 XR 사이의 연결이 필요할 때 이를 중재하는 광역라우팅에 이용된다. 광역라우팅은 뒤에서 다시 설명할 것이다. 광역중재기는 어느 IP 또는 sub-system이 어느 XR에 연결되어 있는가와 몇 개의 XR가 어떻게 연결되어 있는가에 대한 정보를 가지고 마스터와 슬레이브 사이를 연결하는 최적화된 경로를 선택하여 라우팅 정보를 XR에 보내 채널을 형성하도록 한다. 이때 SNA는 다중 채널을 제공하므로 첫 번째 라우팅 경로 중 일부가 점유되어 사용하지 못한 경우에는 다음 경로를 생성하여 XR에 정보를 보낼 수 있다.

스위치 래퍼(switch wrapper: SW)는 IP와 XR 사이에서 신호변환과 직접라우팅(direct routing)을 담당한다. 하나의 스위치 래퍼에는 N=2인 경우 2개의 IP가 연결되고 이 IP들이 SNP를 인터페이스 프로토콜로 이용할 경우에는 신호변환이 필요 없지만 그렇지 않을 경우에는 프로토콜을 변환한다. 현재 AMBA 인터페이스를 갖는 IP가 상당히 많이 존재하므로 그림 2에 나타난 것처럼 AMBA 신호를 SNP 신호로 바꾸는 작업이 필요하다. SW는 또한 XR이나 GA를 거치지 않는 직접라우팅을 지원한다. 이에 대해서는 뒤에서 다시 설명할 것이다. SNA에는 IP 대신 다수의 IP와 지역버스를 포함

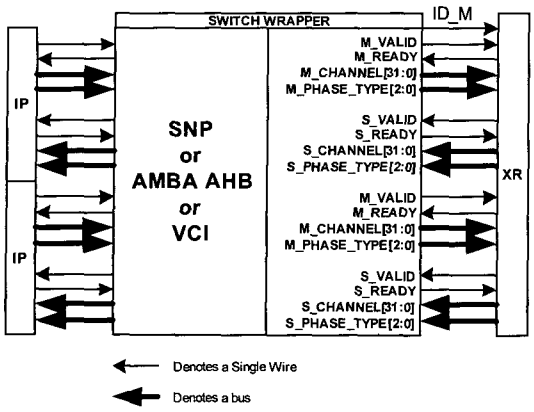

그림 2. Switch Wrapper의 Interface
Fig. 2. Interface of Switch Wrapper.

하는 sub-system이 연결될 수 있다. 특정 IP사이에 빈번한 자료교환이 이루어지는 경우에는 sub-system을 구성하는 것이 효율적이다. 따라서 이 경우 SW는 마스터/슬레이브 역할을 할 수 있는 스위치 브리지로(switch bridge: SB)로 대체된다. SB의 구조는 SW와 거의 같다.

SNA는 직접라우팅, 지역라우팅, 광역라우팅의 3 가지 동작모드를 갖는다. 직접라우팅은 그림 1에서 ①로 표시되어 있으며 SW를 통해서 통신 채널이 형성되어 통신하는 방법이다. 즉, 같은 SW/SB에 연결된 이웃한 IP끼리 통신할 수 있는 전용 통신방법이다. 따라서 XR의 어떤 자원도 사용하지 않고 통신하기 때문에 여분의 통신채널을 확보하는 결과가 된다. 또한 중재를 받지 않기 때문에 중재 잠복기(*arbitration latency*)는 없다. 빈번한 통신이 이루어지는 IP들 같은 경우 동일한 SW에 배치하여 직접라우팅을 통해 데이터를 교환하면 중재 잠복기 없이 통신을 하면서도 추가 통신채널을 확보할 수 있어 버스 이용 효율을 크게 향상시킬 수 있다.

지역라우팅은 그림 1의 ②로 나타낸 것처럼 하나의 XR를 거쳐 동일 XR에 연결된 IP사이의 통신을 의미한다. 지역라우팅에서는 마스터가 라우팅 요청을 하면 XSC가 신호를 받아 목적지가 동일 XR내에 있는지와 채널형성이 가능한 지를 확인하고 XSM에 연결정보를 전달하여 채널형성을 완료한다. 동시에 여러 IP가 요청을 한 경우에는 미리 정해진 규칙에 따라 우선순위가 높은 IP의 요청을 먼저 처리한다. 마스터에서 라우팅을 요청할 때 슬레이브의 주소를 이용하게 되는데 일반적으로 주소는 32 bit 이상의 큰 값이므로 XSC에서 주소값을 보고 슬레이브의 주소를 ID로 변환한 후 라우팅 동작을 진행한다. 마스터의 ID는 SW가 가지고 있고 신호를 XR로 전달할 때 ID를 추가한다. 지역라우팅의 중재 잠복기는 1 사이클이다.

광역라우팅은 그림 1의 ③으로 나타낸 것처럼 두 개 이상의 XR를 통해서 이루어지는 모든 통신을 의미한다. 라우팅 요청을 받은 XR는 목적지가 동일 XR에 없는 경우 광역중재기에 광역라우팅을 요청하게 된다. 이때 마스터와 슬레이브의 ID를 보내면 광역중재기에서는 광역 LUT의 정보를 이용하여 라우팅 경로(routing path)를 결정하고 관련 XR에 연결정보를 제공한다. 정보를 받은 XR는 이웃 XR와의 연결 또는 목적지 IP link와 연결을 시도한다. 이때 광역중재기는 XR사이의 연결에 대해서는 어느 면(side)이라는 정보만을 제공하고 그 면에서 어느 링크를 이용해 채널을 형성할지는 XSC에서 정해진 규칙에 따라 결정한다. 이것은 광역중재기의 부담을 줄이면서 XR 사이의 링크중 하나가 사용되고 있을 경우 남는 링크를 스스로 결정하도록 하여 라우팅 성공 가능성을 높이는 방법이다. 경로중 어느 한 곳이라도 연결에 실패하면 XSC는 광역중재기에 즉시 실패 사실을 알리고 광역중재기는 라우팅 요청을 다른 경로를 선정하여 라우팅을 시도한다. 광역중재기에서 라우팅 정보를 보낼 때 XR에서 지역라우팅 요청을 동시에 받는 경우에는 사용자가 정한 우선순위에 따라 요청을 수행하는데 기본 값은 지역라우팅이 우선순위가 높다. 광역라우팅은 광역중재기를 거치므로 중재 잠복기가 3 사이클이 된다. 따라서 백본(backbone)통신용으로 적합하다.

SNA가 기본적으로 대규모 SoC의 백본버스를 목표로 만들어진 것이지만 IP의 배치를 최적화하여 직접라우팅과 지역라우팅을 많이 활용할 수 있다면 일반 지역버스로도 활용이 가능하다. SNA는 XR를 자유로이 배치할 수 있고 그 수에도 제한이 없으므로 뛰어난 확장성을 가지고 있고 배치방법에 따라 그 특성을 달리할 수 있다. 예를 들어 배치방법에 따라 지역전송이 주가 될 때 효율적인 구조를 만들 수 도 있고 광역전송이 주가 될 때 더 효율적인 구조를 만들 수도 있다.

SNA는 현재 IP와 IP 사이를 직접 연결하는 채널을 제공하는 것으로 되어 있지만 XR를 통해 통신을 하므로 XR에 메모리를 포함시키고 프로토콜을 조금 수정하여 store-and-forward 방식으로 통신을 하도록 할 수 있다. 즉, XR에서 IP에서 보낸 데이터 패킷을 저장하고 다음 XR로 전달하여 최종적으로 목적지에 도착하도록 하는 방식이다. 이것은 SoC의 규모가 매우 커져 XR의 수가 크게 증가하는 경우에 효율적인 통신을 제공하는

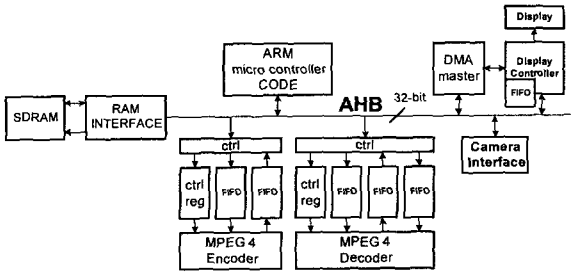

그림 3. Digital Camera 상용제품 블록도[8]
Fig. 3. Block diagram of a digital camera.[8]

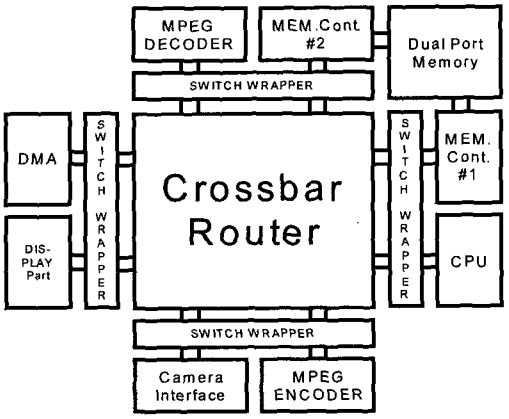

그림 4. 제안된 SNA를 적용한 블록도
Fig. 4. Block diagram of the digital camera with one-XR system.

방법이 된다. 이 경우 잠복기가 증가하고 일정하지 않은 문제점이 있으나 버스 사용 효율은 증가시킬 수 있다. SNA와 NoC와의 비교는 다음 장에서 자세히 다루도록 하겠다.

그림 3은 MPEG4 IP가 내장된 디지털 카메라 제품의 블록다이어그램으로서[8] AMBA AHB 버스를 사용하고 있다. 이 경우 하나의 SDRAM을 DMA, ARM 프로세서, MPEG4 encoder/decoder가 동시에 접근하게 될 경우가 발생한다고 가정한다면 전체 시스템에서 IP 사이의 통신에 상당한 병목이 야기된다. 이러한 문제를 본 논문에서 제시하는 버스구조에 적용시키게 되면 그림 4와 같이 구성할 수 있고 병렬적으로 통신이 가능하기 때문에 평균 40% 이상의 성능개선 효과를 볼 수 있다.

그림 5는 그림 4 보다 좀 더 복잡한 예제에 대한 구조를 보여주고 있다. 즉 그림 4에서 USB, Flash 메모리와 지역버스를 갖는 sub-system까지 3개의 IP가 추가되었다. 이 모델은 제안된 버스에서 광역라우팅을 볼 수 있는 가장 간단한 구조이다. 만약 더 많은 IP가 사용되기를 원한다면 XR를 더 확장하거나 IP 대신 sub-system 버스를 사용하여 확장이 가능하다. XR를 더 많이 확장하면 보다 더 복잡한 시스템을 구성할 수 있다. 시스템을 구성할 때 특정 IP간에 빈번한 통신을

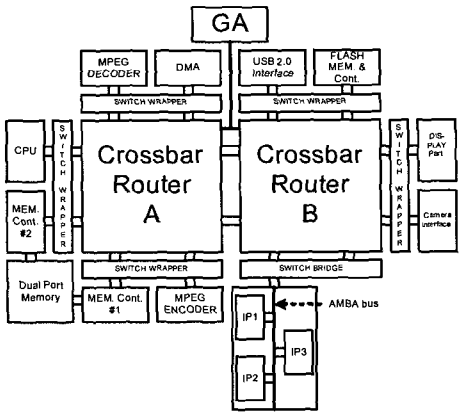

그림 5. 두 개의 Crossbar Router를 적용한 블록도
Fig. 5. Block diagram of the digital camera with 2-XR system.

하는 경우라면 지역라우팅을, 백본(backbone) 버스와 같이 잠복기(latency)가 크게 중요한 곳이 아니며 버스트 전송이 대부분일 경우 광역라우팅 사용을 권장한다.

III. 실험 및 구현

앞 절에서 예시한 그림 4의 구조에 대해서 AMBA AHB와의 성능을 모의실험으로 비교했다. 앞 절에서 언급했듯이 대부분의 기존 SoC 버스가 공유버스 구조이며, 대부분의 특징이 유사하기 때문에 본 논문에서는 AMBA AHB를 비교대상으로 실험을 진행했다. 먼저 실험 시나리오를 구성하기 위해서 몇 가지 제약조건을 가정한다. CPU, MPEG decoder/encoder, DMA는 버스 마스터로서 동작하고 버스트 전송은 16 사이클로 제한한다. 버스 폭은 AHB와 제안된 버스(SNP)가 동일하며, 하나의 슬레이브에 대해서 중첩 요청(overlapping request)을 하지 않는 것을 가정한다. 이것은 분배 전송(split transfer)를 완벽하게 지원해주지 못하는 슬레이브일 경우 어떤 버스에서도 해결할 수 없는 문제로 시뮬레이션 조건을 복잡하게 할 뿐이다. AMBA AHB의 중재 잠복기는 1 사이클로 하며 대기 상태(wait state)는 없다고 가정한다. 또한 모든 과정에서 데이터 처리 시간은 16 사이클로 가정한다. 동작은 다음과 같다.

1. CPU는 memory controller #2로부터 instruction /data를 16 사이클 동안 fetch/read 한다.

2. MPEG encoder는 카메라 인터페이스로부터 16 버스트 전송 동안 입력을 받아 그 데이터를 인코딩한 후 다시 16 버스트 전송 동안 memory controller #1에 저장한다.

3. MPEG decoder는 memory controller #1로부터 16 버스트 전송 동안 데이터를 읽어온 다음 그 데이터를 디코딩한 후 DMA에 16 버스트 전송 동안 전달하고 DMA는 받은 데이터를 16 버스트 전송 동안 display part로 내보낸다.

먼저 AMBA AHB에 대해서 위의 과정에 대해서 수식적으로 비교해보면 표 1과 같다. 표 1에서 AHB의 전체 수행 사이클은 1,377이 되며 공유버스 구조를 가지기 때문에 이 결과 자체가 수행 사이클이 된다. 본 논문에서 제안된 SNA를 적용시키면 모든 작업이 동시에 처리 가능하게 된다. 과정 1에서는 직접라우팅을 사용하기 때문에 크로스바 라우터 A에 버스요청을 하지 않는다. 과정 2에서는 MPEG encoder와 카메라 인터페이스사이에 광역라우팅이 요구된다. 과정 3은 지역라우팅이 요구되며 과정 2와 동시에 실행될 수 있지만 지역라우팅이 광역라우팅보다 우선순위가 높기 때문에 과정 2에서 1 사이클의 지연(delay)이 발생한다. 세 과정이 모두 동시에 진행되기 때문에 전체 수행 사이클은 가장 긴 실행 사이클을 가지는 과정 3의 실행 사이클과 같게 된다. 따라서 전체 수행 사이클은 848이 되며 앞에서 소개한 그림 4의 구조를 적용했을 때는 광역라우팅이 없고 지역라우팅만 있기 때문에 전체 수행 사이클은 816이 된다. 소개한 예제에서 SNA 시스템의 버스 대역폭은 AMBA 시스템과 비교하여 약 40%가 개선된 것을 알 수 있다.

그림 6은 4개의 프로세서에 의해서 멀티프로세싱이 가능한 시스템을 AMBA와 SNA를 이용하여 각각 구성한 것이다. 그림 6(a)의 각 레이어의 프로새서는 독립적

표 1. 그림 3과 4의 예제에 대한 AMBA AHB와 제안된 버스 구조의 동작 사이클 수 비교

Table 1. Comparison of operation cycles for systems in Fig. 3 and Fig. 4.

	AMBA AHB			SNA		
Process	Process 1	Process 2	Process 3	Process 1	Process 2	Process 3
Number of Cycles	17(CPU)	272 (Enc. Read) 16(Process) 272 (Enc. Write)	272 (Dec. Read) 16(Process) 272 (Dec. Write) 272 (DMA Write)	16(CPU)	273 (Enc. Read) 16(Process) 272 (Enc. Write)	272 (Dec. Read) 16(Process) 256 (Dec. Write) 304 (DMA Write)
Total Cycle	1377 Cycles			848 Cycles		

인 프로세싱이 가능하기 때문에 다른 레이어의 프로세서에 의한 동작에 관련된 영향을 받지 않는다. 하지만 각 레이어에서는 공유 SRAM을 가지고 있고 이 메모리는 각 레이어의 프로세서에 의해서 브리지를 통해 접근되어질 수 있다. 또한 각 레이어의 SLAVE는 각 레이어의 프로세서에 의해서만 접근되어 진다. 이러한 구조의 시스템을 제안된 SNA 구조를 이용하여 구성하면 그림 6(b)와 같이 구성될 수 있다. 각 레이어의 슬레이브는 지역성이(local) 강하기 때문에 로컬버스를 구성하여 시스템을 구성하고, 각 SRAM은 지역성이 약하고 광역성이(global) 강하기 때문에 별도의 포트에 연결될 수 있다. 이때 순차적으로 각각의 프로세서가 각 레이어의 SRAM을 B 만큼의 버스트 전송을 하고난 후 자신이 속한 Slave에 다시 B 만큼의 버스트 전송을 하게 된다고 가정한다. 이때 arbitration/bridge latency는 버스트전송에 비해 크게 고려할 부분이 아니기 때문에 무시한다. 이런 경우 그림 6(b)에서는 8B 만큼의 전송 사이클이 필요하며, 그림 6(a)에서는 18B 만큼의 전송 사이클이 필요하다. 이것은 서로 다른 레이어의 프로세서가 다른 레이어의 SRAM을 접근할 때 브리지를 통해서 접근하게 된다. 이때 브리지를 통해서 두 개 이상의 버스가 하나의 통신을 위해 희생되어야 하기 때문에 다른 통신들은 연쇄적으로 delay를 가지게 된다. 즉, 레이어가 증가하면 다른 레이어상에서의 멀티프로세싱을 할 때 성능이 떨어지게 된다. 또한 레이어의 증가할수록 브리지에서 발생되는 latency는 더 이상 무시할 수 있는 값이 아니다. 따라서 위와 같은 멀티프로세싱이 가능한 시스템을 구성하게 되면 최대 약 2배 이상의 성능 차이나 나타남을 시뮬레이션을 통해 확인하였다. 이 때 두 시스템의 면적은 그림 6(b)시스템이 그림 6(a)시스템보다 약 1.7배가량 크다. 그림 6(a)와 같은 방법으로 계속적인 레이어를 확장하게 되면 각 레이어마다 중재기를 포함시켜야하며, 레이어간의 연결을 위해 브리지를 추가해야한다. 이 브리지는 슬레이브와 마스터 두 개의 기능을 모두 포함하고 있어야하기 때문에 그 면적은 전체 구조에서 상당량을 차지하게 된다. 위에서 언급했듯이 레이어의 단순한 증가가 성능에 크게 도움이 되지 않기 때문에 면적의 증가로 성능개선의 이점은 크게 볼 수 없을 것이다. 또한 이러한 복잡한 토폴로지를 사용하게 된다면 시스템을 개발자는 보다 많은 경험을 통해 효율적인 IP 배치에 주의를 기울여야 하기 때문에 개발기간에 많은 영향을 끼칠 수 있다. 그림 6에서는 브리지를 통한 레이어 사이의 통신을 공유 SRAM 접근에만 한정하여 최소화 시켰으나 실제 시스템에서는 이보다

표 2. NoC와 SNA의 특성 비교.
Table 2. Comparison of characteristics of the SNA and NoCs.

	면적	포트 수	프로토콜	버스/제어 비트폭	최대 추정 대역폭	구 현
Marescaux[9]	223 Slice(XCV800) (2,007 Slice)	2 (9)	Custom	16/3	320 Mbits/s(40MHz)	Tristate/Uni-direction
HERMES[10]	316 Slice(XC2V1000) (2,844 Slice)	3 (9)	OCP	8/2	500 Mbits/s(25MHz)	Buffer/Bi-Direction
SNA	830 Slice(XC2V1000)	8	AHB/SNA	32/5	3.2 Gbits/s(25MHz)	Mux./Uni-direction

많은 브리지를 통한 통신이 일어날 것이므로 레이어 증가에 따른 성능 향상 효과는 더욱 큰 차이를 보이게 된다.

표 2에서 제안된 SNA의 구성 요소의 합성 결과와 기존 NoC와의 비교자료를 제시하였다. Marescaux[9] 구조는 포트 수가 2 개 밖에 되지 않기 때문에 더 많은 IP를 위해서는 더 큰 면적이 필요하며, 인터페이스가 정해지지 않고 설계자가 임의로 설계해야하기 때문에 스펙에 따라 영향을 받을 수 있다. 또한 Tri-state를 사용하므로 개발자에 의한 테스트가 쉽지 않다. 한편, HERMES[10] 구조는 포트 수가 3 개인데 실제 하나의 모듈로는 네트워크를 구성할 수 없기 때문에 더 많은 모듈이 필요하다. 이 구조 역시 양방향 전송을 기반으로 한 구조이기 때문에 자동화된 검증이 어려우며, 버퍼로 인하여 transfer latency가 발생한다. 세 구조의 최대 추정 가능한 대역폭의 차이는 6-10 배 정도로 SNA가 앞서 있다고 할 수 있다. 또한 IP 확장성이나 테스트 면에서 더 나은 점을 볼 수 있으며, 면적에서는 약 2.4-3.5 배 정도의 차이가 난다. 그러나 동일한 포트수로 맞추면 NoC의 면적이 더 커짐을 알 수 있다. 제안된 SNA는 기종의 SoC 온칩버스를 확장하여 대체하는 것으로 기본적으로 NoC와는 지향하는 목표와 접근 방법에 차이가 있기 때문에 두 구조를 직접적으로 비교하는 것은 무리가 있다.

한편 SNA는 point-to-point 프로토콜인 SNP를 이용하면서 하나의 채널이 크로스바 라우터를 거쳐 IP 사이를 연결하고 항상 일정한 채널이 형성되므로 시스템의 크기가 커져도 연결선에 의한 부하가 크게 늘어나지 않는다. 또한 선의 수가 기존 공유버스에 비해 작아 버스에 의한 부하가 기존 버스보다 작으므로 전력 소모를 줄일 수 있는 여지가 있다. 반면 AMBA 버스는 IP 수가 늘어날수록 구동해야하는 부하가 증가하고 라우팅도 복잡해져 전력소모가 급격히 증가한다. 또한 IP에서 버스 인코딩을 해주어야 하는 AMBA 시스템과는 달리 SW에 버스 인코딩 회로를 내장시켜 IP에서는 관여하지 않아도 버스 신호 변환을 줄일 수 있어 추가 전력

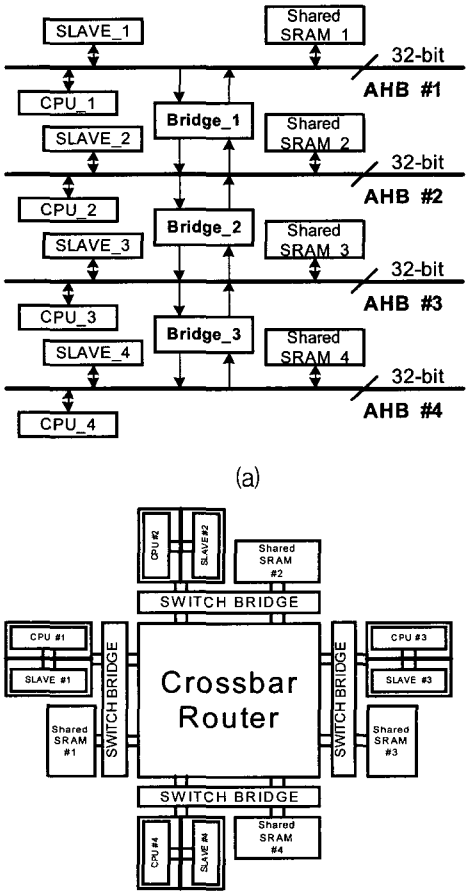

그림 6. 멀티프로세싱 시스템 (a) AMBA AHB 4 Layer를 이용한 구조 (b) 단일 XR를 이용한 SNA 구조
Fig. 6. Multi-processor system (a) AMBA AHB 4 layer architecture (b) SNA system with a XR.

소모를 줄일 수 있다. 따라서 비교적 저전력 구현이 가능하다.

SNA의 또 다른 특징은 단방향을 버스를 사용하므로 들어오고 나가는 버스의 연결선을 서로 교대로 배치하여 사용하지 않는 방향의 선들을 접지시킴으로써 상호 신호 간섭을 배제하여 고속 신호 전달에 적합한 라우팅이 가능하다는 점이다. 또한 필요할 경우 차동신호를 이용할 수 있어 고속 및 저전력 구현이 가능하다.

Ⅳ. 결 론

본 논문에서 SoC에서 효율적인 IP간의 통신을 위한 새로운 다중 채널 SoC 버스구조인 SNA(SoC Network Architecture)를 제안한다. SNA는 AMBA AHB와의 호환성을 유지하면서 버스 효율성을 높일 수 있는 SNP를 인터페이스 프로토콜(Interface Protocol)로 사용하였다. 제안된 SNA는 SW(Switch Wrapper) 또는 SB(Switch Bridge), XR(Crossbar Router), GA(Global Router)로 구성된다. 하나의 XR는 최대 4개의 다중 채널을 지원 가능하며 시스템 확장이 용이하다. 제안된 SNA는 3개의 동작모드를 가지며 연결의 성격에 따라 구분될 수 있다. 제안된 SNA 구조는 모델시스템에 대한 시뮬레이션 결과에서 AMBA 버스보다 40% 개선된 성능을 보였다. 또한 상대적으로 적은 면적을 가지면서 유연한 확장성을 가지므로 효율적인 SoC 내부통신을 제공할 수 있다.

참 고 문 헌

[1] Inside the New Computer Industry, issue 138, Jan 2001.

[2] F. MORAES, N. CALAZANS, et al., "HERMES: an Infrastructure for Low Area Overhead Packet -switching Networks on Chip", Integration, the VLSI Journal (accepted for publication)

[3] P. Gurrier, A. Greiner, "A Generic Architecture for On-Chip Packet-Switched Interconnections", Proceedings of the conference on Design, Automation and Test in Europe, Paris, France, pp. 250 - 256, 2000.

[4] ARM, "AMBA Specification, Revision 2.0", 1999.

[5] W. Peterson, "WISHBONE SoC Architecture Specification, Revision B.2", Silicore Corporation, 2001.

[6] IBM, "CoreConnect Bus Architecture", 1999.

[7] Jaesung Lee, Hyuk-Jae Lee, and Chanho Lee, "SNP: A New Communication Protocol for SoC", in Int'l Conf. on Communications, Circuits and Systems, June 2004.

[8] Hatro Products OY, "http://www.hantro.com/pdf/overview.pdf"

[9] Marescaux, T.; Bartic, A.; Verkest, D.; Vernalde, S.; Lauwereins, R. "Interconnection Networks Enable Fine -Grain Dynamic Multi-Tasking on FPGAs", In: Field-Programmable Logic and Applications (FPL'02), pp. 795-805, Sep. 2002.

[10] Moraes, F., Mello, A., Möller, L., Ost, L., Calazans, N., "A Low Area Overhead Packet -switched Network on Chip: Architecture and Prototyping", IFIP Very Large Scale Integration (VLSI-SOC). pp. 318-323, Dec. 2003.

저 자 소 개

이 상 헌(학생회원)
2003년 숭실대학교 전자공학과 학사졸업.
2003년~현재 숭실대학교 전자공학과 석사재학.
<주관심분야 : SoC 설계, On-Chip-Network/Bus 연구>

이 혁 재(정회원)
1987년 서울대학교 전자공학과 학사졸업.
1989년 서울대학교 대학원 전자공학과 석사졸업.
1996년 미국 퍼듀대학교 전기컴퓨터공학과 박사졸업.
1996년~1998년 루이지애나 공과대학 컴퓨터공학과 조교수.
1998년~2001년 인텔 선임연구원.
2001년~현재 서울대학교 전기컴퓨터공학부 부교수
<주관심분야 : 컴퓨터 아키텍처 및 멀티미디어용 SOC 설계>

이 찬 호(정회원)
1987년 서울대학교 전자공학과 학사졸업.
1989년 서울대학교 대학원 전자공학과 석사졸업.
1994년 University of California, Los Angeles 전자공학과 박사졸업.
1994년 8월~1995년 2월 삼성전자 반도체연구소 선임연구원.
1995년 3월~현재 숭실대학교 정보통신전자공학부 부교수.
<주관심분야 : 채널코덱의 VLSI 구현, SoC on-chip-network, SoC 설계 방법론, 저전력 프로세서 설계>